CLAIMS

What is claimed is:

1. A method of executing a computer algorithm, comprising:

   executing a first module encapsulating said computer algorithm except at least one communication operation of said algorithm;

   executing a second module encapsulating said at least one communication operation of said algorithm, such that said at least one communication operation is available to said first module.

2. The method of claim 1, wherein said at least one communication operation comprises at least one environment-dependent communication operation of said algorithm.

3. The method of claim 2, wherein said at least one environment-dependent communication operation comprises all environment-dependent communication operations of said algorithm.

4. The method of claim 1, further comprising executing a third module encapsulating another communication operation of said algorithm.

5. The method of claim 1, further comprising instantiating at least one data object for encapsulating data communicated between said first module and a communicating partner, each one of said at least one data object being an instance of a data class, said data communicated between said first module and said communicating partner being accessible by said first module.

6. The method of claim 5, wherein data from said first module is encapsulated in a first data object being an instance of a first data class, and data to said first module is encapsulated in a second data object being an instance of a second data class.

7. The method of claim 6, wherein said second module comprises a communication object, said communication object being an instance of a communication class.

8. The method of claim 7, wherein said first module comprises a command object, said command object being an instance of a command class.

9. The method of claim 8, wherein each one of said classes implements one of a plurality of protocols of a framework, such that instances of said classes are compatible with each other.

10. The method of claim 9, wherein said framework is a Java framework and each one of said plurality of protocols is respectively encapsulated in an interface.

11. The method of claim 10, wherein said command class implements a command interface, said command interface defining at least one method of executing, said method of executing taking an indicator of said communication object as a parameter, thereby an operation of said communication object is available to said command object.

12. The method of claim 11, wherein said communication class implements a communication interface, said communication interface defining at least one method of communication.

13. The method of claim 12, wherein said at least one method of communication comprises a method of communicating data from said first data object to said communicating partner.

14. The method of claim 13, wherein said at least one method of communication comprises a method of communicating data from said communicating partner to said second data object.

15. A computer readable medium storing thereon computer executable instruction code, said code when executed by a processor of a computer causes said processor to:

execute a first module encapsulating a computer algorithm except at least one communication operation of said algorithm;

execute a second module encapsulating said at least one communication operation of said algorithm, such that said at least one communication operation is available to said first module.

16. The computer readable medium of claim 15, wherein said at least one communication operation comprises at least one environment-dependent communication operation of said algorithm.

17. The computer readable medium of claim 16, wherein said at least one environment-dependent communication operation comprises all environment-dependent communication operations of said algorithm.

18. The computer readable medium of claim 15, wherein said processor is further caused to instantiate at least one data object for encapsulating data communicated between said first module and a communicating partner, each one of said at least one data object being an instance of a data class, said data communicated between said first module and said communicating partner being accessible by said first module.

19. The computer readable medium of claim 18, wherein data from said first module is encapsulated in a first data object being an instance of a first data class, and data to said first module is encapsulated in a second data object being an instance of a second data class.

20. The computer readable medium of claim 19, wherein said second module comprises a communication object, said communication object being an instance of a communication class.

21. The computer readable medium of claim 20, wherein said first module comprises a command object, said command object being an instance of a command class.

22. The computer readable medium of claim 21, wherein each one of said classes implements one of a plurality of protocols of a framework, such that instances of said classes are compatible with each other.

23. The computer readable medium of claim 22, wherein said framework is a Java framework and each one of said plurality of protocols is respectively encapsulated in an interface.

24. A method of encapsulating a computer algorithm, comprising:

encapsulating, in a first module code, said computer algorithm except at least one communication operation of said computer algorithm;

encapsulating said at least one communication operation in a second module code;

so that one of said first and second module codes can be modified without changing the other one of said first and second module code.

25. The method of claim 24, wherein said at least one communication operation comprises all environment-dependent communication operations of said algorithm.

26. The method of claim 24, wherein each one of said first and second module codes implements a common protocol so that said first and second module codes are compatible.